VŠB - TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

DIPLOMOVÁ PRÁCE

2010 Bc. Karel Mozdřeň

VŠB - TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY
KATEDRA INFORMAČNÍCH TECHNOLOGIÍ

Detekce mokré vozovky z jedoucího vozidla na základě obrazové analýzy

Wet Road Detection from a Moving Vehicle Using Image Analysis

2010 Bc. Karel Mozdřeň

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

podpis: .....

Abstrakt a klíčová slova

Tato diplomová práce pojednává o metodě detekce mokré vozovky na základě analýzy obrazů získaných z kamery zabudované v automobilu. V úvodu seznamuje čtenáře s problematikou detekce mokré vozovky, aktuálním stavem této problematiky a nastiňuje navrženou metodu detekce. V další části vysvětluje jednotlivé kroky detekce a pro pohodlí čtenáře i teoretický základ k pochopení netriviálních částí algoritmu. V posledních kapitolách jsou uvedeny výsledky experimentů a shrnutí všech poznatků získaných v průběhu práce.

analýza obrazu, mokrá vozovka, Kamera, automobil, OpenCV, C++

Abstract and key words

This diploma thesis introduces a method for wet road detection that is based on analyzing the images taken by a car inbuilt camera. At the beginning, it provides the reader with basic understanding the wet road detection problems, state of the art, and it also proposes the basic principles of the designed detection method. Further, it gives the reader the full step-by-step understanding of the method. For convenience of the reader, also the subchapters are included that make understanding the non-trivial theoretical parts easier. In the last chapter, the experimental results are presented.

Computer Vision, Image Analysis, Wet Road, Camera, Car, OpenCV, C++

Obsah

1 Úvod

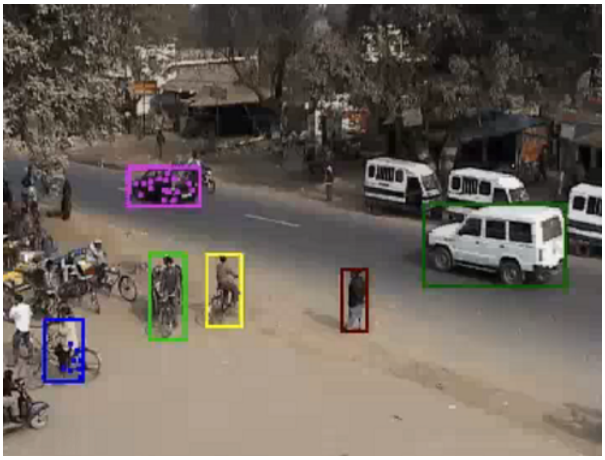
Zpracování a analýza obrazu si našla uplatnění již v mnoha odvětvích průmyslu. Má mnohá využití například při automatickém měření vyrobených výrobků, jejich rozpoznávání, klasifikace objektu a mnoho dalších, která většinou souvisí s robotikou. Zpracování a analýza obrazu se nasazuje většinou tam, kde není možné postavit člověka, nebo je lehce nahraditelný strojem. Má poskytnout takový pohled na věc, kterého člověk sám o sobě není vůbec schopen.

Poslední dobou se mnoho výzkumných pracovníků zabývá problematikou inteligentních transportních systémů(ITS). To jsou systémy, které mají asistovat člověku při přepravě lidí nebo věcí obecně. Hlavním tématem jsou asistenční systémy. Tyto systémy se dají rozdělit na dva základní typy. Ty, které jsou postaveny mimo přepravní vozidlo, a ty, které jsou ve vozidle zabudovány. Zabudované systémy mají za úkol sledovat okolí vozidla a vnější zase okolí jisté oblasti, do které jsou nasazeny. Jejich kombinací vzniká systém, jenž neposkytuje informaci pouze řidiči jednoho vozidla, ale také umožňuje informovat ostatní řidiče o aktuální situaci.

Tyto systémy se v neposlední řadě zasazují o bezpečnější přepravu. Existuje mnoho aplikací takových systémů. Vnější asistenční systémy jsou často nasazovány tam, kde je doprava nějakým způsobem komplikovaná a je třeba ji kontrolovat automaticky. Na hlavních dálničních tazích se často používají k měření hustoty provozu. Kontroluje se rychlost jednotlivých vozidel a provádí se jejich identifikace. Na křižovatkách zase může sledovat plynulost provozu, nebo detekovat překážky a kolize.

Obrázek 1: Systém určený ke kontrole vozidel na dálnici

Vnitřní asistenční systémy naopak mají napomáhat řidiči přímo v souvislosti s řízením. Častými aplikacemi jsou kontrola bdělosti řidiče sledováním

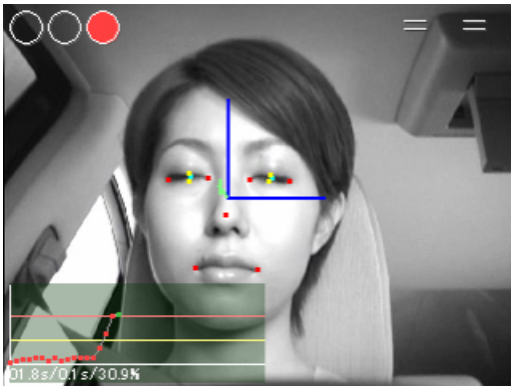


Obrázek 2: Sledování provozu v Indii

Obrázek 3: Sledování provozu v tunelu

jeho očních víček, nebo také sledování krajnice silnice. Přejede-li ji řidič, bývá systémem na tuto skutečnost upozorněn.

Jednou komponentou takového systému může být rozpoznávač mokré vozovky. Jak víme, automobil je na mokré vozovce mnohem hůře ovladatelný a vzniká tak bezpečnostní riziko. Nejlepší prevencí je na riziko včas řidiče upozornit, aby mohl přizpůsobit jízdu stavu vozovky. Tato práce pojednává o konstrukci takového detektoru, jeho testování, uvádí použité technické prostředky a nezbytnou teorii, která je zde uvedena pro pohodlí čtenáře. V závěru sumarizuje výsledky a poskytuje pohled na další možnosti řešení a jeho vylepšení.



Obrázek 4: Kontrola bdělosti řidiče od fy. Nissan

Obrázek 5: Ukázka možného vstupu a výstupu detektoru

2 Současný stav

V současné době existuje jen velmi málo publikací na téma rozpoznávání mokré vozovky. Je to dáno převážně tím, že jde o poměrně novou oblast zkoumání a její řešení nejsou jednoduchá. Osobně mě zaujaly tři publikace, popisující metody rozpoznání mokré vozovky, se kterými jsem se při průzkumu současného stavu setkal. Pokusím se nastínit princip metod popisovaných v těchto publikacích.

2.1 Metoda klasifikace mokré/suché vozovky pomocí odražených a incidenčních světel pozorovaných na snímcích získaných z jedné kamery

Metoda je stručně popsaná v publikacích [Tes08]. V této metodě se předpokládá 3D tvar prostředí, který se dá modelovat pomocí tří ploch. Celá

metoda se skládá ze tří kroků. Nejprve spočítáme prostorový pomocný obraz tak, že získáme obrázky pohledu na silnici shora. Poté se spočítá incidenční světlo pro každou oblast. Dále se oblasti klasifikují na tři druhy: mokrá, suchá, nerozhodnutelná.

Pokud jde o odražená světla, mokrá vozovka odráží mnohem více světla než suchá, ta je pouze šedá po celém svém povrchu a odražené světlo není tak závislé na úhlu pohledu. Data o intenzitě odraženého světla získáme z jedoucího automobilu. Snímky jdoucí za sebou vytvářejí prostorový obraz, takže můžeme sledovat každý bod z různých úhlů pohledu. Mohou nastat dva případy, v jednom má bod spekulární složku a v druhém se buď intenzita nezměnila, nebo ji nemá.

Incidenční světlo spočítáme z okolního prostředí. Předpokládáme, že můžeme doslova převrátit celé prostředí na oblast silnice a získat tak simulované zrcadlení. Nakonec se použijí tři prahy k určení do jaké třídy daná oblast patří.

2.2 Detekce mokré oblasti vozovky založené na saturovaném odrazu

Jde o publikaci [Tes07] od stejných autorů jako v předešlé metodě. Zde se využívá planární projekční matice(homografie). Skládá se ze dvou základních algoritmů. První algoritmus měří rychlost, kterou vozidlo jede a druhý měří přenesenou intenzitu odrazu světla z vozovky.

Homografie se používá k získání obrazu povrchu vozovky aplikováním projekční matice na obraz získaný z kamery. Předpokládáme přitom, že známe vztah mezi kamerou a vozovkou. Dá se říct, že tak získáme pohled na vozovku shora.

Měření rychlosti odpovídá problému hledání odpovídajících obrazů. Respektive jejich vzájemné translaci a rotaci. Zde se zase předpokládá, že se v oblasti nepohybuje žádný jiný objekt.

Takto získané obrazy jsou poté identické, pokud se překryjí. Sledujeme poté změnu intenzity odrazu ve všech snímcích. Pokud je vozovka mokrá, mění se intenzita velmi často a hodně, pokud je však suchá, zůstává uniformní(nemění se). Využíváme tedy Lambertova modelu. Posledním krokem je použití prahování k označení mokrých a suchých oblastí.

2.3 Detekce mokré vozovky s použitím polarizačních filtrů

V další metodě [Ran09] se používá polarizačních filtrů. Využívá vlastností polarizace odraženého světla. Na suché vozovce se odráží vertikálně i horizontálně polarizované světlo stejně. Na mokré vozovce se horizontálně polarizované světlo při odrazu utlumuje. Získáme tedy dva obrazy, každý s využitím jiného polarizačního filtru, a sledujeme rozdíly. Na obrazech se suchým povrchem bude rozdíl malý. Na mokrých bude daleko zřetelnější [6].

Obrázek 6: Ukázka vlivu polarizace světla odraženého od mokrého povrchu snímaného kamerou

3 Řešení

Mým úkolem je zpracovat detektor, který bude schopen nalézt na vozovce mokré oblasti. K tomuto účelu mám použít jednu kameru umístěnou na předním skle automobilu. Nalezené oblasti budou na výstupním obraze zřetelně označeny. To vše s maximální možnou spolehlivostí, které jsem schopný dosáhnout.

Mé řešení bude vycházet z metod, které byly prezentovány ve zmíněných publikacích, kromě metody využívající polarizačních filtrů, protože požaduje více než jednu kameru. Jelikož však neexistuje detailní popis těchto metod, musím přijít sám na postupy a algoritmy, které vedou ke kýženému cíli. Hotové řešení následně osobně odzkouším a naměřené hodnoty úspěšnosti a rychlosti detekce uvedu na konci dokumentu.

3.1 Kamerový subsystém

Kamera jako taková by měla být malá a schopná získat obraz s dobrou kvalitou a rozlišením. Pro tento účel jsem vybral barevnou průmyslovou kameru uEye2230-C německé firmy IDS-Imaging s rozlišením 1024×768 a frekvencí až 30 snímků za sekundu [obr. 7]. Počet snímků za sekundu se může zvýšit, pokud použiji menší než maximální možné rozlišení. Tato kamera je dostupná i v monochromatické verzi (uEye2230-M).

Kamera se umisťuje na přední sklo vozidla na stranu spolujezdce [obr. 8]. Držák kamery je provizorní a v praxi by měl být nahrazen sofistikovanějším mechanizmem. Na hrbolaté cestě se může stát, že se obraz třese, ale třes je v mezích. V případě, že bych chtěl pracovat na rozpoznávači založeném na polarizačním filtru, není problém zapojit do systému kamery dvě. Toho mohu využít i v případě, že chci provést trojrozměrnou rekonstrukci. Ke kameře jsou na internetu volně ke stažení ovladače, knihovny a různé nástroje a to i pro linux.

Obrázek 7: Barevná kamera uEye2230 s objektivem H0814-MP

Obrázek 8: Uchycení kamery na předním skle, na straně spolujezdce

3.2 OpenCV

K implementaci jsem zvolil s výhodou OpenCV, což je balíček otevřených knihoven pro zpracování a analýzu obrazu. Většina používaných operací je zde implementována se srozumitelným rozhraním a efektivně. Kromě toho je OpenCV multiplatformní a běží tak spolehlivě na několika operačních systémech. A hlavně, je běžně používaný pro aplikace ve zpracování a analýze obrazu.

3.3 Navrhované řešení

Už jsem zmínil, jakou kameru a které knihovny použiji k implementaci rozpoznávače, ale pořád nebyla zmíněná funkce rozpoznávače. Za prvé je třeba určit, co je vstupem a co výstupem rozpoznávače.

Vstupem rozpoznávače je obraz z kamery zabudované v automobilu. Kamera je připevněna na přední sklo na straně spolujezdce a snímá obraz ve

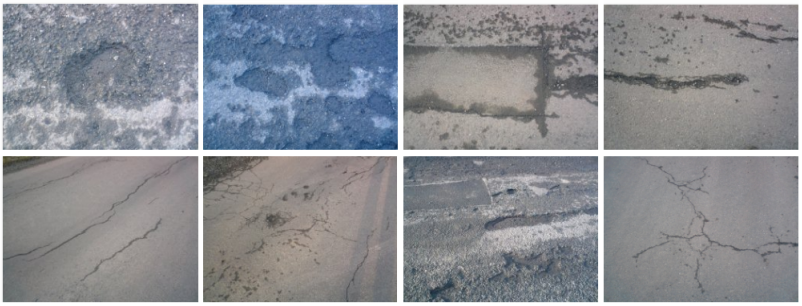
směru jízdy automobilu [obr. 9]. Na tomto obrazu mě hlavně zajímá cesta. Vše ostatní prozatím zanedbávám. Povrch cesty má našedivělou barvu a její barva se většinou nemění. Na cestě se mohou vyskytovat obrazce malované bílou barvou, které fungují jako orientační prvky. Označují například krajnici, středovou čáru, nebo šipky znázorňující, ve kterém odbočovacím pruhu se vozidlo pohybuje [obr. 10]. Kromě těchto značek můžeme pozorovat občasné nedostatky, jako jsou praskliny a díry. Tyto objekty jsou zřetelné i za špatného počasí a lze je tedy také použít jako orientační body [obr. 11].

Obrázek 9: Pohled z kamery připevněné na přední sklo

Obrázek 10: Některé orientační prvky na vozovce

Kromě těchto objektů se na povrchu vozovky objevují tmavá místa, která jsou často mokrá nebo jenom zastíněná. Ve většině případů lze rozlišit stín od mokré oblasti sledováním odstínu šedi. Mokrá vozovka bývá často mnohem tmavší než stín. Navíc pokud se pohybujeme v prostoru a sledujeme oblast, kterou považujeme za mokrou, tak u zastíněné části se nebude měnit intenzita odraženého světla tak moc jako u oblastí mokrých. U těch se mnohem víc projevuje zrcadlová složka.

Horší je situace u oblastí, které nejsou příliš mokré. Dá se o nich tvrdit, že jsou pouze navlhlé a zrcadlení není tak zřejmé, nebo se vůbec neproje-



Obrázek 11: Některé nedokonalosti povrhu vozovky

vuje. V takovém případě mohu stín rozlišit pomocí analýzy barvy. Jsou-li body obrazu zapsány modelem RGB(červená, zelená, modrá), má zastíněná oblast stejně intenzivní složky R(Červená) a G(Zelená), ale složka B(Modrá) je silnější. Většina systémů však pracuje pouze s obrazem černobílým, což je mnohem rychlejší než zpracovávat obraz barevný, neboť jsme nuceni pracovat jen s jednou barevnou složkou namísto tří. Proto je také pravděpodobnější, že bych k rozpoznání stínu sledoval spíše změny intenzit nežli barevné rozdíly. U zastíněných oblastí je změna intenzit odstínu šedi stejně malá jako u nezastíněných suchých oblastí vozovky.

Detekce stínu nebude pro systém tak kritická, jelikož trochu navlhlá vozovka je téměř tak bezpečná jako vozovka suchá. Z hlediska analýzy rizik je ovšem bezpečnější detekovat suchou oblast jako mokrou (stín je detekován jako mokrá vozovka), nežli mokrou vozovku jako suchou. Nejzřetelnějším prvkem mokré vozovky jsou kaluže. Kaluž se může zdát jako zrcadlo položené na cestu. Téměř dokonale odráží veškeré světlo, které na její hladinu dopadá. Zároveň ji můžeme považovat za nejnebezpečnější prvek. K jejich detekci je vhodné tuto vlastnost sledovat.

Podařilo se mi zachytit všechny zmíněné případy najednou na jedné fotografii [obr. 13]. Ve středu obrazu můžeme pozorovat stín, který vrhla moje postava jak na suchou tak na mokrou vozovku, a v levé části vozovky je zřetelný odraz světla od vozovky (zrcadlení). Odráží se v něm bílý dům.

Obrázek 12: Odraz oblohy v kaluži

Obrázek 13: Mokrá, suchá, zastíněná a velice mokrá vozovka v jedné fotografii

3.3.1 Sledování změn intenzity barvy z různých úhlů pohledu

Oblast vozovky musí být sledovaná z více úhlů pohledu, abychom odhalili tu mokrou. Jednoduchým řešením by bylo použití systému více kamer. Každá kamera by sledovala oblasti z jiného úhlu a detekce by ve většině případů byla pouze o nalezení rozdílu. S jednou kamerou takovou možnost nemám. Můžu však sledovat změny v čase. Pohybuje-li se automobil a kamera snímá obraz, lze říct, že jde o sledování oblasti z různých úhlů pohledu. Abych však nalezl sobě si odpovídající oblasti, musím nejdříve být schopen získat povrch vozovky transformací z prostorového obrazu. Výsledkem této transformace bude obraz povrchu vozovky připomínající pohled shora [obr. 14].

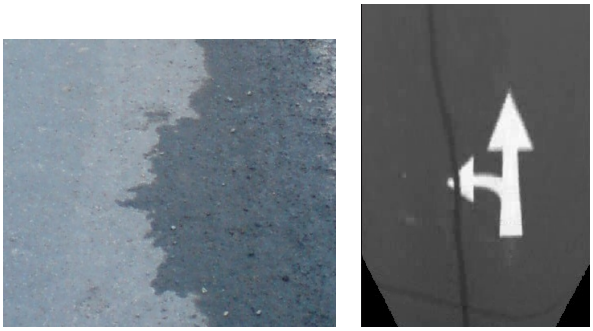
Když se automobil pohybuje, je pak obraz v každém transformovaném snímku mírně posunutý a natočený. K zjištění velikosti úhlu natočení a posunu musím znát body zájmu, pomocí kterých můžu provést korespondence, tzn. nalézt sobě odpovídající si body. Pomocí nich si spočítám translaci a rotaci a obrazy přeložím přes sebe. Sledováním rozdílů intenzit sobě si odpovídajících bodů obrazů pak určím, zda je vozovka mokrá, nebo suchá. Nevýhodou tohoto zpracování je však potřeba mít neustále k dispozici záchytné body. Ne vždy se podaří takové body nalézt a musím si tak pomoct jinak. Řešením tohoto problému je použít nějaký filtr, který je schopný předpovědět, jak moc se další snímek posune a otočí. Nejznámějším filtrem, který to dokáže, je Kalmanův filtr. Za jeho pomoci jsem schopen v případech, kdy nemám záchytné body, odhadnout posun a zajistit tak spojitost řešení.

3.3.2 Sledování navlhlé vozovky

U navlhlých oblastí, kde se změna intenzity v čase tolik neprojeví, se použije prahování. Vychází se z předpokladu, že mokrá vozovka je mnohem tmavší než vozovka suchá [obr. 14]. Problém je tu však s nastavením prahu. Jak určit práh, aby co nejlépe rozděloval vozovku na suchou a mokrou? Nemáme jinou možnost než provést dostatek měření. Až bude nastavený práh podle mých představ, smím jej dále používat. Případně mohu využít adaptivního prahu, jenž mění prahovou hodnotu podle celkového vjemu obrazu.

3.3.3 Reflexivní mokrá vozovka

Posledním typem oblastí mokré vozovky, které lze rozpoznat, jsou kaluže. Myšlenka detekce kaluže, jakožto zrcadla vozovky, je relativně jednoduchá. Položíte-li zrcadlo na zem a popojdete kousek dál od tohoto zrcadla, tak v něm uvidíte převrácený obraz za ním. Nejlépe je tento jev pozorovatelný na velkých stojatých vodních plochách, kde je obraz nad horizontem zobrazený převrácený pod horizontem[15].

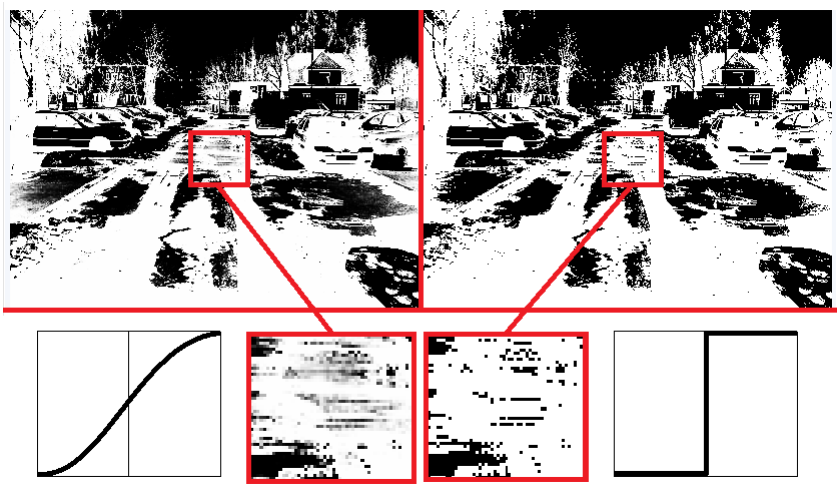


Obrázek 14: Ukázka navlhlé vozovky a pohledu shora na vozovku

Obrázek 15: Velká vodní plocha s odrazem prostředí za ní

Ve skutečnosti se mokré oblasti nedají jednoznačně rozdělit na navlhlé, mokré a kaluže. Obyčejně se prolínají, a tak i navrhované metody detekce nedetekují pouze jeden typ, ale jejich výstupy se prolínají. My však máme za úkol označit na výstupu oblasti mokré vozovky. To znamená, že musím výstupy nějakým způsobem spojit. Jako první možnost se nabízí logické operace disjunkce z binární logiky. Pokud by alespoň jeden z detektorů určil, že je vozovka v daném místě mokrá, byla by oblast na výstupu označená jako mokrá, i kdyby ostatní detektory nic nedetekovaly. Toto řešení je však příliš striktní. Použiji-li však pro všechny moduly Fuzzy logiku, mohu tvrdit se stupněm pravdivosti od 0 do 1, že je vozovka mokrá. Nad těmito operacemi opět provedeme disjunkci. Ta je definovaná jako maximální hodnota. Bude tak počítat celkový stupeň pravdivosti pro danou oblast. To ovšem znamená, že ve všech modulech, které k detekci použijeme, nemůžeme vracet binární obrazy na výstupech. Veškeré ostré prahování pomocí funkce signum nahradím prahováním hladší funkce jako je například sigmoid [obr. 16] . Konečný výstup pak mohu případně již zpracovat ostrým prahováním. Myslím si však, že hladší prahovaný obraz bude pro výstup vhodnější už

jenom proto, že si jej můžeme představit jako pravděpodobnost, že je vozovka mokrá.



Obrázek 16: Porovnání prahování: vlevo sigmoid, vpravo signum

3.4 Algoritmus

Než začnu podrobně rozebírat celý detektor, je vhodné, abych shrnul jednotlivé kroky jeho funkce. Kroky jsou rozepsány postupně za sebou v pořadí, ve kterém se provádějí.

1. Nejprve získám obraz z kamery, nebo videozáznamu v případě, že neprovádím detekci v reálném čase. Obraz převedu z barevného na odstíny šedi a případně odstraním rušivé vlivy, jako je například šum.

2. Dále musím najít oblast vozovky. Jelikož má kamera zafixovanou pozici, mohu si určit oblast tam, kde se vozovka nachází s největší pravděpodobností. Oblast lze určit parametry, jako jsou: úběžník (místo kde končí silnice v obraze), ohnisková vzdálenost (pro prostorovou transformaci) a parametry near a far pro nastavení vzdálenosti hranic sledované oblasti.

3. K dalšímu zpracování se použijí tři moduly. Každý detekuje jeden typ mokré vozovky.

   – Modul pro intenzitu světla (vlhká vozovka)
   Zde pomocí prahování určím, zda je vozovka mokrá, nebo suchá. S využitím funkce sigmoida získám plynulý přechod a tím i stupeň pravdivosti, určující, nakolik jsem si jistý, že je vozovka mokrá.

- Modul reflexe (kaluže)
  V tomto modulu hledám podobnosti nad a pod horizontem. Čím více jsou si obrazy podobné, tím je větší pravděpodobnost, že je v oblasti kaluž.
- Modul sledování změn intenzity odraženého světla v čase (středně mokrá vozovka)
  Nejprve se oblast sledovaného obrazu převede pomocí transformace na pohled shora. Dále se sledují za sebou jdoucí snímky. Mezi těmito snímky se provádí obrazová korespondence, abych mohl obrazy posunout a otočit tak, aby se překrývaly. Poté mohu pozorovat rozdíly v intenzitách světla a určit pomocí prahování, zda jde o vozovku mokrou, nebo o vozovku suchou. Tento výstup se nakonec promítne pomocí zpětné transformace do stejného prostoru, ve kterém pracují ostatní detektory.

5. Spojením výsledků všech modulů pomocí Fuzzy logické operace disjunkce získám celkový výsledek. Ten mohu buď opět prahovat již pomocí standardního prahování na binární obraz, nebo jej poskytnout přímo jako výstupní fuzzy logický obraz.

4 Detailní rozbor řešení

4.1 Převod barevného obrazu na černobílý

Nejprve je třeba obraz trochu předzpracovat, než na něm budeme provádět nějaké detekce. Vstupem celého systému je obraz barevný. Každý bod obsahuje tři složky: červenou, zelenou a modrou. Podle velikosti každé ze složek se bude vytvářet výsledná barva daného bodu. Bude-li například červená složka mnohem větší než složky ostatní, bude se barva bodu blížit více k červené. Stejně tak to platí pro všechny ostatní.

Při zpracovávání obrazu se většinou pracuje s celým obrazem. Pokud jde o obraz barevný v systému RGB, musíme každou operaci provést pro každou složku samostatně. Proto se ve většině případů na barvu zapomíná a obraz se převádí na odstíny šedi. Pouze v málo případech je potřeba zpracovávat obraz barevný, neboť většina informace v obraze převedeném z barevného na odstíny šedi zůstává zachována.

Převod se provádí pomocí vzorce (1). Jak moc se bude která složka podílet na výsledné intenzitě, je určeno pomocí spektrální křivky citlivosti. Není nijak přesně určeno, v jakém podílu to přesně bude, protože je to závislé na použitém snímači. Nejčastěji jsou to však hodnoty koeficientů ve zmíněném vzorci.

$$I = 0.3I_R + 0.59I_G + 0.11I_B \tag{1}$$

Ukázka převodu z barevného obrazu na obraz v odstínech šedi je na obrázku [obr. 17]. Samotný obrázek je důkazem toho, že se žádné závažné informace neztratily. Stále je možné rozpoznat všechny objekty a tvary.

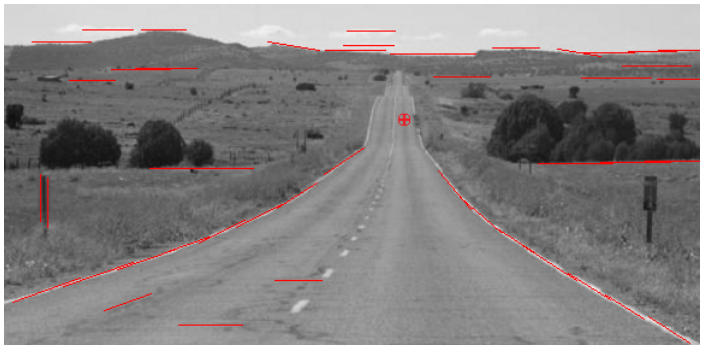
Obrázek 17: Ukázka převodu obrázku z RGB na odstíny šedi

4.2 Získání povrchu vozovky z obrazu

Prvním krokem k detekci mokré vozovky je samotné nalezení oblasti vozovky. Hlavním úkolem je pak získat reprezentaci jejího povrchu. Není to

zrovna triviální úkol, ale existuje několik postupů, jak dojít úspěšně k nalezení požadované oblasti. Některé z nich tady uvedu a jeden vyberu.

Jedním takovým způsobem je použití detektoru úseček(přímek). Detektor spustíme a sledujeme ty úsečky, které reprezentují krajnice vozovky. Těchto úseček však bude několik. Spočteme průsečíky jednotlivých úseček a zprůměrujeme jejich souřadnice do jednoho bodu. Ten bude označovat bod do ztracena (úběžník). Ukazuje přibližně místo, kde na horizontu končí vozovka. Z úseček na levé krajnici, pravé krajnici a tohoto bodu jsem schopný sestrojit jednoduchý trojúhelník, jenž znázorňuje hledanou oblast vozovky. Ukázku aplikace detektoru úseček je možné pozorovat na obrázku [obr. 18].



Obrázek 18: Detekované úsečky v obraze a naznačený bod do ztracena

Další možností je sledovat zájmové body v několika za sebou jdoucích snímcích a pomocí korespondence obrazu vytvořit spojením sobě odpovídajících bodů přímky. Na těch pak opět mohu hledat průsečíky a znovu průměrováním jejich souřadnic nalézt úběžník. V tomto případě ale nebude úběžník určen čistě vozovkou, ale směrem jízdy automobilu.

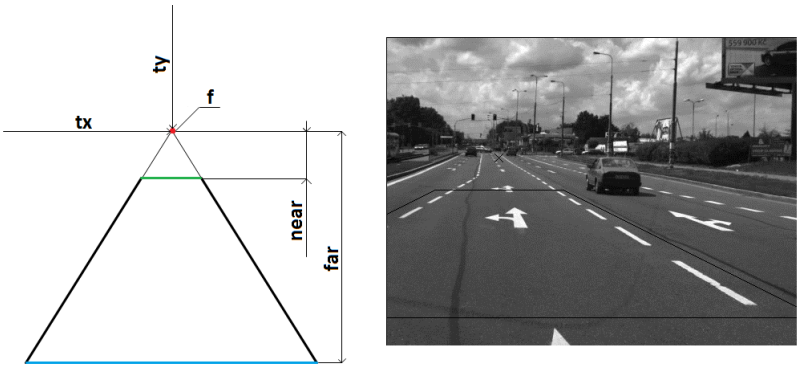
Kombinací obou metod by mimochodem bylo možné vytvořit něco jako autopilota do automobilu. kde máme směr, ve kterém jede automobil, a směr, ve kterém ubíhá vozovka. Každopádně obě metody jsou výpočetně relativně náročné a nedoporučuji tedy použití ani jedné z nich. A navíc u obou metod hrozí, že se nepodaří nalézt úběžník a celý výpočet by tak přišel nazmar.

Nejrychlejším a zároveň nejjednodušším řešením je ruční určení sledované oblasti vozovky. Stačí si uvědomit, že je kamera zafixována na předním skle automobilu a řidiči se zpravidla více či méně daří udržet vozidlo ve směru ubíhání vozovky. Kamera se natáčí zároveň s tímto vozidlem, a proto se s vysokou pravděpodobností nachází sledovaná vozovka na jednom místě v obraze. Mně pak stačí pomocí nastavení několika parametrů tuto oblast určit. Parametry, které se nastavují jsou:

- **f** - ohnisková vzdálenost kamery

- **tx** - x-ová souřadnice bodu do ztracena v obraze
- **ty** - y-ová souřadnice bodu do ztracena v obraze
- **far** - zadní hranice snímaného prostoru
- **near** - přední hranice snímaného prostoru

Předpokladem je, že cesta je zpravidla plochá a je pak možné získat její povrchovou reprezentaci pomocí jednoduché homogenní transformace, do níž tyto parametry zadám. Na obrázku [obr. 19] pak ukazuji význam všech těchto parametrů.



Obrázek 19: Vstupní parametry programu pro získání povrchu silnice a jejich význam ve zpracovávaném obraze

Výsledná transformační matice pak bude složena z matice středové projekce a translace, viz. vzorec (2). Tato transformace bude transformovat souřadnice bodů výsledného obrazu (plochy vozovky) na souřadnice z obrazu kamery. Přečte si na těchto souřadnicích barvu a vloží ji na vstupní souřadnice. To mi zaručí, že v obraze povrchu vozovky nebudou díry, které by vznikly při přímém zobrazování po pixelech z obrazu kamery, kde je velký úsek vozovky znázorněn pouze několika málo pixely.

$$T_1 = \begin{pmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{pmatrix}, T_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & f & 0 \end{pmatrix}, x' = T_1 T_2 x \qquad (2)$$

Samotná transformace poté vytváří nový obraz, který je jistou aproximací pohledu shora na vozovku, jak je možné pozorovat na obrázku 20.

Obrázek 20: Vlevo vstupní obraz s naznačenou oblastí snímání a vpravo výstupní obraz po transformaci

4.3 Vlastnosti mokré vozovky

Než si naznačíme, jak přesně detektor bude fungovat, je třeba se trochu víc zabývat tím, jaké vlastnosti má mokrá vozovka z pohledu pozorovatele. Jak jsem již uvedl, rozděluji mokrou vozovku na tři základní druhy: vlhká, středně mokrá a kaluže. Vlhká vozovka a kaluže jsou celkem jasně rozlišitelné. Vlhká vozovka je zpravidla tmavší než vozovka suchá a kaluže se zase v obraze projevují jako zrcadla položená na vozovku zobrazující otočený obraz za ním [obr. 21].

Obrázek 21: Ukázka navlhlé vozovky a kaluže

Mnohem obtížnější je rozpoznání středně mokré vozovky. V tomto případě se nemůžeme spolehnout na to, že je vozovka tmavší než vozovka suchá, protože se zde již projevuje zrcadlení v takové formě, že téměř kopíruje barvu objektu za ní. Nedá se však ani říct, že by bylo zrcadlení tak výrazné, že by se v něm daly přímo rozpoznat objekty.

Obrázek 22: Ukázka středně mokré vozovky

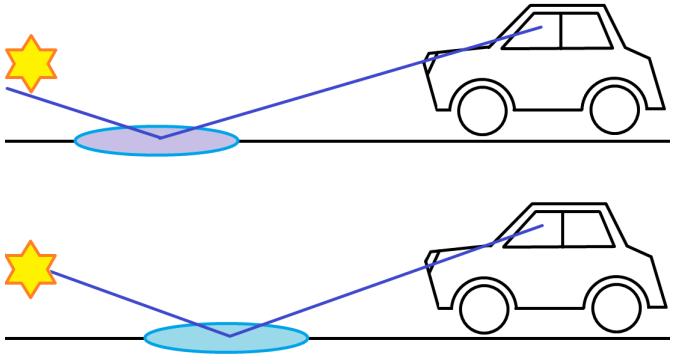
K nalezení takové oblasti se využívá netriviálního postupu sledování změny intenzity svítivosti bodu vzhledem k úhlu pohledu na vozovku. Abych však byl schopný sledovat nějakou oblast vozovky z několika úhlů, musím mít k dispozici několik kamer. To ale odporuje mému zadání, a tak musím postupovat jinak.

Řešení mého problému se prakticky nabízí samo. Jednou kamerou je možné také sledovat oblasti z různých úhlů pohledu. Pohybuje-li se automobil směrem vpřed, získám na jednu oblast z každého snímku jiný pohled, jak napovídá obrázek [Obr 23]. Jak je možné si všimnout, jsou to nejen pohledy posunuté, ale i pod rozdílným úhlem. Světlo, které se pak odráží od vozovky do kamery, se tímto vlivem změní. A změna intenzity tohoto odraženého světla je hodnota, podle které určuji, zda je vozovka mokrá, nebo suchá. Suchá vozovka na rozdíl od mokré vozovky odráží světlo uniformně a ke změnám intenzity tam nedochází.

Teoreticky, kdybych měl dvě kamery, nejen že bych měl rovnou dva rozdílné pohledy na jednu oblast vozovky, ale mohl bych použit i polarizační filtry. Světlo, odražené od mokré vozovky, je totiž polarizované (na rozdíl od světla odraženého od vozovky suché) a sledováním této polarizace je jsem schopen jednoduše rozeznávat [obr. 24]. Na obrázcích jsou dva pohledy na mokrý povrch bláta. Jednou s a jednou bez polarizačního filtru.

4.4 Návrh systému detekce oblastí mokré vozovky na základě analýzy jejich vlastností

Žádná ze zmíněných vlastností není natolik určující, že bych čistě na její analýze byl schopen rozpoznat veškeré mokré oblasti vozovky. Mnohem lepším řešením se jeví využít všech vlastností a na základě jejich analýzy detektor postavit. Jak ale spojit detektory jednotlivých vlastností tak, aby



Obrázek 23: Sledování mokré vozovky z různých úhlu pohledu jedoucího automobilu

Obrázek 24: Ukázka vlivu polarizace světla odraženého od mokrého povrchu snímaného kamerou vlevo bez a vpravo s polarizačním filtrem

se mi ve výsledku ukázal celkový obraz mokrých oblastí?

Pokud beru v potaz binární logiku, pomocí které je možné zpracovat pouze dva stavy(mokrá/suchá), tak použiji operaci disjunkce($u \vee v$). Použití této logiky však velice omezuje. Neumožňuje například určit, s jakou pravděpodobností tento detektor určí správně oblast mokré vozovky, nebo s jakou jistotou můžeme tvrdit, že je oblast opravdu mokrá. Tento problém řeší takzvaná Fuzzy logika.

4.4.1 Fuzzy logika

Fuzzy logika umožňuje pracovat nejen s hodnotami *pravda/nepravda*, ale také s mnohem jemnějším modelem, jako je například {určitě ne, možná ne, nevím, možná ano, určitě ano}. Logické hodnoty v tomto případě nenabývají pouze hodnot $1, 0$, ale jde o čísla $\forall x \in \Re : 0 \leq x \leq 1$. Pokud proměnná nabude hodnoty 1, mám 100% jistotu, že je hodnota pravdivá. Naopak čím

více se hodnota blíží k 0, tím mám větší jistotu, že jde o nepravdivý výrok. V případě, že je hodnota rovna 0.5, není výsledek vůbec jistý (patří do obou skupin zároveň). Tyto hodnoty tedy vyjadřují stupeň pravdivosti.

Abych mohl pracovat se stupni pravdivosti, musím si nejdříve nad nimi definovat základní logické operátory, které se ve Fuzzy logice nazývají Zadehovy operátory podle autora. Definují se tři základní operace: negace (komplement) (3) , disjunkce(4) a konjunkce(5). Jsou definovány takto:

$$\bar{u} = 1 - u \tag{3}$$

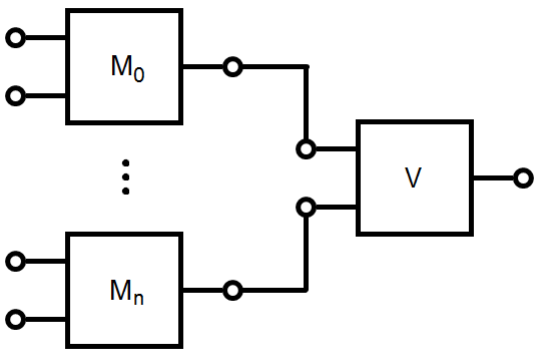
$$u \vee v = max(u, v) \tag{4}$$

$$u \wedge v = min(u, v) \tag{5}$$

Fuzzy logiku lze jednoduše pochopit na příkladu sklenice. Ve standardní výrokové logice je možné sklenici zařadit mezi prázdné, nebo plné. Fuzzy logika dovoluje určit příslušnost k oběma množinám. Naliji-li do litrové sklenice 0.7l mléka, bude patřit do množiny plné se stupněm pravdivosti 0.7 a do množiny prázdné se stupněm pravdivosti 0.3.

4.4.2 Moduly detektoru

Jak již bylo dříve řečeno, je třeba při detekci vzít v potaz více faktorů. Každý modul bude mít svůj vlastní vstup a výstup. Důležitý je jednotný výstup, který bude udávat stupeň pravdivosti $0 \leq y_n \leq 1$. Výstupem celého systému bude disjunkce výstupů všech modulů [obr. 25].



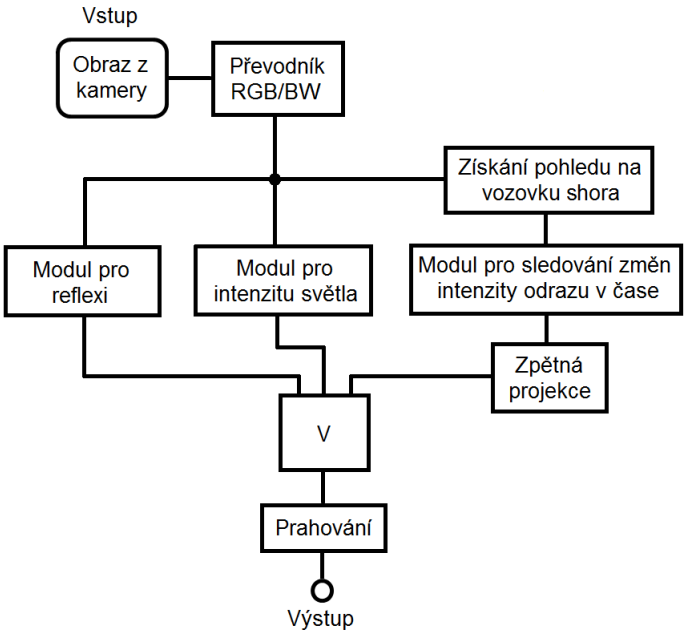
Obrázek 25: Obecný pohled na moduly a jejich propojení

V mém případě bude výstupem každého modulu obraz, který bude obsahovat hodnoty v rozsahu 0..255, které budou reprezentací rozsahu 0..1. Logické operace se pak budou provádět pro každý bod ve všech obrazech.

Proto také vytvořím odpovídající logické operace nad obrazy (konjunkce a disjunkce). Výstupem systému tedy bude taktéž obraz s rozsahem hodnot 0..255. Výstup bude generován třemi základními moduly:

- **Modul pro intenzitu světla** - vstupem bude obraz povrchu vozovky.
- **Modul reflexe** - vstupem bude obraz získaný z kamery.
- **Modul sledování změn intenzity odraženého světla v čase** - zde je vstupem sekvence za sebou jdoucích snímků povrchu vozovky.

Výstup každého modulu můžeme navíc omezit nebo zesílit konstantou, která bude určovat, nakolik budu výstupu důvěřovat. Bude-li tato konstanta nastavena například na hodnotu 0.8, nemůže žádný stupeň pravdivosti tuto hodnotu překročit. Ovšem pokud překročí hodnotu 1.0, bude se jí výstup násobit(zesilovat). Nepřesáhne však hodnotu 255.0. Rozvržení celého systému pro detekci mokré vozovky je nakresleno na následujícím obrázku [obr. 26].



Obrázek 26: Pohled na celý systém a všechny jeho moduly

4.5 Modul sledování změn intenzity odraženého světla v čase

Nyní je čas podívat se podrobně na všechny moduly. Nejprve uvádím ten nejsložitější. Jde o modul, který je určený k rozpoznání středně mokré vo-

zovky. Jde o jediný typ vozovky, který nelze rozeznat jediným pohledem. K rozpoznání tohoto typu mokré vozovky je třeba ji sledovat z několika úhlů. Jak jsem uváděl v předchozích částech textu, toho dosáhneme sledováním několika za sebou jdoucích snímků. V těchto obrazech pak sledujeme změnu intenzity bodů.

Tyto snímky se musí nejprve upravit do mého požadovaného pohledu shora. Získám tak snímky povrchu vozovky, které jsou vzájemně trochu posunuté, případně i natočené. Abych byl schopný sledovat změny intenzit, musím nejdříve najít sobě odpovídající body v obrazech. Pro nalezení sobě si odpovídajících bodů tedy musím najít takovou transformaci, která mi jeden obraz posune na druhý.

Pro nalezení této transformace jsem zkusil několik postupů. Jedním z nich bylo zkoušení posunutí obrazu do všech možných směrů tak dlouho, dokud nebyly rozdíly minimální. To však vedlo k velice dlouhým časům. Zkoušel jsem dále porovnávat pouze malé oblasti obrazu, což sice nalezení transformace urychlilo, ale docházelo také k nalezení nesprávných parametrů transformace z důvodu častých podobností několika vzájemně posunutých, sobě si neodpovídajících oblastí obrazů.

Nejvhodnějším postupem je nalezení korespondence obrazu za pomoci výrazných bodů v obrazech. Tuto detekci provedu v obou po sobě jdoucích obrazech. Poté sleduji okolí těchto výrazných bodů a porovnávám je vzájemně. Ty, u kterých jsou rozdíly nejmenší, považuji za sobě si odpovídající. K nalezení výrazných bodů jsem použil Harrisův detektor rohů. Je rychlý a relativně přesný.

4.5.1 Harrisův detektor rohů

Harrisův detektor rohů a hran [Har88] vychází z Moravcova detektoru [Mor80]. Sleduje se v něm změna rozptylu s posunem v souřadnicích obrazu. Využívá se zde malého okna, které je typicky velikosti 3×3 - 7×7 pixelů. Tímto oknem se posunuje ve směru vertikálním, horizontálním a nakonec po obou diagonálách. Změna je zde tedy vypočtena jako suma diferencí intenzit odpovídajících pixelů umocněných dvěma ze dvou vzájemně posunutých oken (rovnice 6).

$$E = \sum_{u,v} W_{u,v} \left(I\left(m+u, n+v\right) - I\left(u,v\right)\right)^2 \tag{6}$$

Aplikováním tohoto Moravcova operátoru na celý obraz se vytvoří mapa odezvy rohů (rohovost) . Poté se v této mapě může provést prahování. Nakonec se hledají lokální maxima v $min(E)$, které označují hledané rohy.

Je potřeba si uvědomit, že Moravcův operátor pracuje pouze v omezeném úhlu, například 45°. To způsobuje při hledání korespondence dvou obrazů, které jsou vzájemně pootočené, problémy (nedetekují se rohy na stejných

místech). To znemožňuje jeho použití v mém řešení. Tento problém řeší Harrisův detektor rohů. Využívá se zde autokorelace. Jednoduchou analýzou lze zjistit, že se všechny malé uhlové posuny dají zapsat přepisem předchozího vzorce (6) na následující vzorec (7).

$$E = \sum_{u,v} W_{u,v}(mM + nN + O(m^2, n^2))^2 \tag{7}$$

Kde M a N ve vzorcích (8),(9) naznačují výpočet prvního gradientu.

$$M = I \otimes (1, 0, -1) \approx \frac{\delta I}{\delta m} \tag{8}$$

$$N = I \otimes (1, 0, -1)^T \approx \frac{\delta I}{\delta n} \tag{9}$$

Jelikož jde pouze o malé posuny, lze E zapsat takto(10):

$$E(m, n) = Am^2 + 2Cmn + Bn^2 \tag{10}$$

$$\begin{aligned} A &= M^2 \otimes W \\ B &= N^2 \otimes W \\ C &= (MN) \otimes W \end{aligned}$$

Protože odezva rohů může být narušena šumem, používá se jako okenní funkce funkce Gaussova(11).

$$W_{u,v} = e^{-\left(\frac{u^2+v^2}{2\sigma^2}\right)} \tag{11}$$

Tento operátor má však příliš velkou odezvu na hrany. To, co chci, jsou pouze rohy. Musí se přeformulovat měření odezvy rohu tak, aby brala v potaz směr posunu okna. Což se dá zapsat pro malé posuny takto (12):

$$E(m, n) = (m, n)M(m, n)^T \tag{12}$$

Zde M je symetrická matice 2×2 tvaru (13) :

$$M = \begin{pmatrix} A & C \\ C & B \end{pmatrix} \tag{13}$$

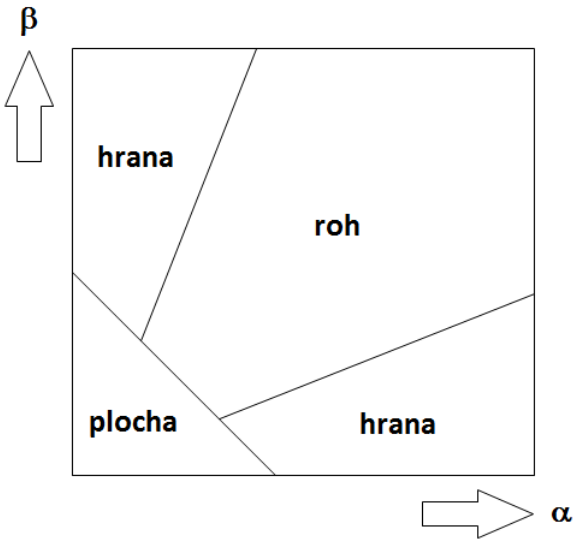
Je možné si všimnout, že E má blízkou spojitost s autokorelační funkcí, kde M popisuje její tvar a směr (z Taylorova rozvoje). Řekněme, že α a β jsou vlastní čísla, která odpovídají zakřivení autokorelační funkce a jsou také racionálními invariantními deskriptory M. Pak se nám nabízejí tyto možnosti:

A. Obě zakřivení jsou malá, takže autokorelační funkce je plochá, pak jsou intenzity v okně relativně konstantní.

B. Když je jedno zakřivení velké a druhé malé, znamená to, že E se mění pouze při pohybu oknem jen jedním směrem. Nalezli jsme tak hranu.

C. Obě zakřivení jsou velká a to znamená, že autokorelační funkce je špičatá a pohyb v okně bude měnit E při pohybu okna všemi směry. Pak jsme nalezli roh.

Všechny tyto případy pro koeficienty α, β lze sledovat na obrázku 27.



Obrázek 27: Rozdělení prostoru pro koeficienty α, β

Nyní je možné přímo přistoupit k měření odezvy rohů a hran, a to invariantně k rotaci. Vytvoří se funkce, která využívá pouze koeficientů α a β. Použitím $Trace(M)$ a $Det(M)$ se vyhneme ruční dekompozici M a spočteme odezvu rohu R pomocí následujících vzorců (14):

$$Trace(M) = \alpha + \beta = A + B \quad (14)$$

$$Det(M) = \alpha\beta = AB - C^2 \quad (15)$$

$$R = Det(M) - kTrace^2(M) \quad (16)$$

k, ve vzorci (16), je empirická konstanta, která se nastavuje na hodnoty kolem $0.04 - 0.06$.

4.5.2 Korespondence obrazů

Když už vím, jak najít zájmové body v obraze, není problém nalézt stejný bod, posunutý v obraze jdoucím za ním, jak bylo popsáno v publikaci [Soj00] v kapitole o analýze obrazu proměnných v čase. Mám k dispozici obrazy x_t a x_{t+1}. Na obou obrazech provedu detekci rohů pomocí Harrisova operátoru a hledám vzájemné rozdíly. Rozdíly se hledají jen v omezené oblasti kolem nalezeného bodu. To znamená, že se opět použije okenní funkce, přibližně o velikosti 10×10. Čím bude okno větší, tím bude měření přesnější, ale zároveň pomalejší. Měření provádím pomocí vzorce (17).

$$E_{Err} = \sum_{u,v} W_{u,v}(I_t(m_1 - u, n_1 - v) - I_{t+1}(m_2 - u, n_2 - v))^2 \tag{17}$$

Významné body, nalezené pomocí Harrisova operátoru, se setřídí podle velikosti odezvy hrany od největšího po nejmenší. Abych nalezl transformaci, která určí o kolik se posunula a otočila vozovka v obraze, použiji jen několik nejvýznamnějších bodů. Minimálně však tři, abych byl schopen určit kromě posunu i rotaci. Pro tyto body z x_t najdu odpovídající body v obraze $x_t + 1$ tak, že sleduji vypočtené rozdíly a ty, jež je mají nejmenší, považuji za sobě si odpovídající. Po nalezení těchto bodů mohu hledat transformaci, která bude obsahovat parametry translace a rotace, což odpovídá transformační matici T (18) v dvojrozměrném homogenním prostoru (18).

$$T = \begin{pmatrix} cos(\alpha) & sin(\alpha) & tx \\ -sin(\alpha) & cos(\alpha) & ty \\ 0 & 0 & 1 \end{pmatrix} \tag{18}$$

Transformační matice má tři neznámé parametry: tx, ty, α. K jejich výpočtu použiji zápis transformační rovnice (19).

$$x' = Tx \tag{19}$$

V této rovnici za x' dosazuji nalezené body z obrazu x_{t+1} a za x body z obrazu x_t. Těchto parametrických rovnic je přesně tolik, kolik párů bodů se použije k výpočtu transformace. Dále stačí řešit soustavu rovnic o třech neznámých. Souřadnice nalezených bodů často nebudou přesné, a proto také nemohu řešit soustavu standardním způsobem. Došel bych k rovnicím, které nemají řešení. Takové rovnice se řeší pomocí předeterminovaných systémů.

4.5.3 Předeterminované systémy

Předeterminovaný systém lineárních rovnic je takový systém, kde počet rovnic je větší než počet proměnných. V mém případě to znamená, že k výpočtu použiji více než tři body. Jelikož nemohu hledat řešení přesné,

hledám alespoň takové řešení, které nejlépe odpovídá všem proměnným. Uvažuji tedy obecnou lineární rovnici tvaru (20) :

$$Ax = b \tag{20}$$

Pak chyba od požadovaného řešení bude r a bude se rovnat rozdílu přesného řešení od řešení nalezeného.

$$r = Ax - b \tag{21}$$

Nejdůležitější je nalézt řešení, v němž bude tato chyba minimální. Hledáme tedy takové r, které bude minimální (22) .

$$min(r^T r) = min((Ax - b)^T (Ax - b)) \tag{22}$$

Pomocí tzv. zobecněné inverze získám tento zápis

$$x^* = (A^T A)^{-1} A^T b, \tag{23}$$

kde x^* nejlépe odpovídá hledanému řešení. Chyba r pak bude nejmenší, pokud x nabude takových hodnot x^*, pro které je vektor r kolmý k vektoru Ax. Kolmost Ax^* a r lze vyjádřit pomocí skalárního součinu.

$$(Ax^*)^T (Ax^* - b) = 0 \tag{24}$$
$$(x^*)^T (A^T Ax^* - A^T b) = 0 \tag{25}$$

Pro řešení je zajímavá pouze druhá závorka levé strany rovnice (25). Rovnice (26) se pak použije k nalezení optimálního řešení (23) .

$$A^T Ax^* - A^T b = 0 \tag{26}$$

4.5.4 Zjednodušení transformace

Práci si mohu zjednodušit tím, že zanedbám rotaci. V takovém případě nás bude zajímat pouze posun v souřadnicích x a y. Tvar matice se tak značně zjednoduší (27) .

$$T = \begin{pmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{pmatrix} \tag{27}$$

K tomu, abych mohl sledovat vzájemné posunutí obrazu, již nemusím provádět složité výpočty a stačí mi pouze odečítat vektory bodů, které si odpovídají. Výsledný vektor určuje samotnou transformaci. Spočtu-li několik vektorů z nejspolehlivějších bodů, mohu následně hledat vektor průměrů, který bude odpovídat hledané transformaci nejlépe (nepotřebuji tedy metodu

řešení předeterminovaných soustav lineárních rovnic) . Avšak zanedbáním rotace se dopouštím vědomě chyby, která se projeví nejvíce, pokud bude automobil zatáčet. Tuto transformaci tedy mohu použít spolehlivě hlavně pro jízdy, kdy se jede převážně rovně s minimálním počtem prudkého zatáčení.

Teď když mám k dispozici parametry transformace, mohu obrazy překládat přes sebe tak, abych byl schopen sledovat změny v intenzitách bodů povrchu vozovky na odpovídajících oblastech. Může se však stát, že jsou detekovány špatně korespondence bodů, ať už kvůli podobnosti, nebo by jednoduše nebyly detekovány žádné, nebo málo významných bodů. V takovém případě musím použít filtr, který umožní takové nesprávné parametry transformace odhalit a poskytnout nápravu. Proto tento účel je vhodný Kalmanův filtr.

4.5.5 Kalmanův filtr

Kalmanův filtr je množina matematických rovnic, která umožňuje efektivní výpočet odhadu stavu, ve kterém se proces nachází, a minimalizuje chyby. Filtr je velice užitečný z několika důvodů: umožňuje odhadnout minulé, aktuální a budoucí stavy a to i v případě, že nezná úplný model systému.

Kalmanův filtr vychází z obecného problému odhadování stavu $x \in \Re^n$ řízeného procesu v diskrétním čase postaveném na stochastické lineární diferenciální rovnici (28)

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{28}$$

a rovnici měření (29)

$$z_k = Hx_k + v_k \tag{29}$$

Náhodné proměnné w_k a v_k reprezentují šum procesu a měření. Předpokládá se, že jsou vzájemně nezávislé, bílé (bílý šum) a mají normální rozdělení. Čtvercová matice A velikosti $n \times n$ v diferenciální rovnici (28) je relací stavu v předchozím časovém kroku $k-1$ k aktuálnímu stavu k s absencí řídícího i procesního šumu. Je třeba poznamenat, že A se opět může měnit v každém kroku, ale také se považuje za konstantní. Matice B velikosti $n \times l$ odpovídá relaci volitelného řídícího vstupu $u \in \Re^l$ ke stavu x. Matice H velikosti $m \times n$ v rovnici měření (29) je relací mezi stavem a měřením z_k. H se také může měnit při každém kroku, a přesto se také považuje za konstantní. Detailnější popis a derivace je možné nalézt v publikaci [Wel06].

Kalmanův filtr používá určitou formu zpětné vazby. Odhaduje stav v jistém čase a získává zašumělá měření jako odezvu. Díky toho existují dva typy rovnic: časové a měřicí aktualizační rovnice. Časové jsou zodpovědné za dopředné promítání (v čase) odhadu v dalším kroku za pomoci kovariance chyby odhadu a aktuálního stavu. Měřicí jsou naproti tomu zodpovědné za zpětnou vazbu (vylepšovat pomocí nových měření odhad budoucího stavu).

Časové aktualizační rovnice mohou být také chápány jako "predikční rovnice" a měřicí aktualizační rovnice jako "korekční rovnice". Algoritmus predikce a korekce pak funguje jako cyklus, ve kterém se střídá odhadování a měření chyby se vzájemnou interakcí.

Časové aktualizační rovnice:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \quad (30)$$
$$P_k^- = AP_{k-1}A^T + Q \quad (31)$$

Měřící aktualizační rovnice:

$$K_k = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1} \quad (32)$$
$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - H\hat{x}_k^-\right) \quad (33)$$
$$P_k = \left(I - K_k H\right) P_k^- \quad (34)$$

$\hat{x}_k$ Odhad dalšího stavu

z_k hodnota aktuálního měření

$H\hat{x}_k^-$ Předpovídaná hodnota měření

P_k^- Kovariance chyby odhadu

K Kalmanův zisk (Váha důvěryhodnosti)

R Kovariance chyby měření

Q Kovariance chyby procesu

Na váhu K (Kalmanův zisk) lze pohlížet tak, že čím více se kovariance chyby měření R blíží k 0, tím více se věří aktuálnímu měření z_k a méně předpokládanému měření $H\hat{x}_k^-$. Na druhou stranu, čím více se kovariance chyby odhadu P_k^- blíží k 0, tím méně se věří aktuálnímu měření a tím více se věří odhadu $H\hat{x}_k^-$.

Než se Kalmanův filtr použije, musí se nastavit parametry. První na řadě je kovariance chyby měření R. Tato hodnota se získá měřením. Není problém získat pár vzorků na filtrovaném procesu a vypočítat chybu.

Mnohem složitějším úkolem je získání kovariance chyby procesu Q. Na rozdíl od kovariance chyby měření R se kovariance chyby procesu Q nedá přímo změřit, jelikož není většinou přístup ke kompletnímu vnitřnímu modelu systému. Její nastavení vyžaduje delší pozorování.

Nejčastějším způsobem nastavení filtru je postupné ladění. To znamená nastavování parametrů R a Q tak, aby dávaly co možná nejlepší výsledky. K tomu se v některých případech používá další Kalmanův filtr, což nazýváme identifikací systému.

V případě, že R a Q jsou konstantní, chyby kovariance odhadu P_k a Kalmanův zisk K_k se rychle stabilizují a zůstanou také konstantní. Tento postup lze použít k odhadu těchto parametrů. Mnohem častějším případem je však proměnlivost parametrů a v takovém případě je jenom na uživateli, jak si parametry nastaví podle pozorování dynamiky systému.

4.5.6 Alternativa ke Kalmanovu filtru

V případě, že se zdá Kalmaův filtr zbytečně složitý, nebo jeho nasazení není zcela nezbytné, je možné přemýšlet nad použitím některé alternativy. V teorii řízení se používá několik postupů, jak filtrovat vstup. Jedním z nich je použití obyčejného průměru. To ale znamená pamatovat si několik posledních měření a daný průměr počítat ze všech těchto naměřených hodnot. Zajímavější je však vytvářet pouze odhad tohoto průměru. Jednou takovou metodou je průměrování s exponenciálním zapomínáním.

Označme číslo n jako reprezentaci počtu hodnot, které se zahrnují do výpočtu. Tyto hodnoty samotné se neukládají. Ukládá se však průměrná hodnota y_avg, která je v prvním kroku rovna první naměřené hodnotě. Každé nové měření x_k má váhu $\frac{1}{n}$. Poslední hodnota, která se do výpočtu nezahrnuje, se označuje jako y_{last}. Její váha bude také $\frac{1}{n}$. Jelikož se poslední hodnota neukládá, aproximuje se pomocí průměru y_{avg}. Výpočet odhadu průměru se pak provádí pomocí vzorce (35) .

$$y_{avg} = y_{avg} + \frac{x_k - y_{avg}}{n} \tag{35}$$

Pro predikci následujícího stavu se předpokládá, že je roven průměrné hodnotě. Předpokladem je tedy, že se další měření bude od průměru lišit jen minimálně.

$$y_{avg|k+1} = y_{avg|k} \tag{36}$$

4.5.7 Vstupní parametry filtru

Obecně vstupem filtru budou parametry nalezené transformace určující pohyb obrazu vozovky v čase. Jednodušší pro pochopení vstupu je sledovat chování auta na vozovce. Auto na vozovce může jet určitou rychlostí, nebo může zrychlovat (couvání nebo zpomalení jsou stejná chování, pouze se zápornou hodnotou). Proto vstupem filtru budou vektory aktuální rychlosti a zrychlení. Známe-li rychlost a zrychlení, můžeme odhadnout relativně přesně, kde se bude auto (vozovka) nacházet v příštím kroku. Rychlost lze vypočítat jako rozdíl vektorů pozic v časech k a $k-1$ (37) , což jsou vlastně parametry transformace.

$$v_k = x_{k-1} - x_k \tag{37}$$

Zrychlení je pak rozdíl rychlostí v těchto časech (38) .

$$a_k = v_{k-1} - v_k \tag{38}$$

Pokud by se sledovala pouze rychlost a zrychlení by bylo zanedbáno, odhady by se mohly při změnách rychlosti hodně lišit od skutečnosti, což by mohlo způsobit chyby.

4.5.8 Výpočet výstupu modulu

Již mám k dispozici parametry transformace, která mi určují, jak se obrazy vzájemně posunuly a otočily. A mohu proto jednoduše zjistit, které body v obou obrazech si vzájemně odpovídají. Výstupem modulu pak bude rozdíl v hodnotách intenzit pro každý bod v obraze.

Poslední krok tohoto modulu je pak převedení výstupu z pohledu shora na pohled z kamery pomocí inverzní transformace k transformaci na pohled shora.

4.6 Modul pro sledování intenzit v obraze

Nyní popíšu další modul, který jsem vyvinul. Jde o modul pro sledování intenzit bodů v obraze. Je určen pro sledování navlhlých oblastí silnice. To jsou ty, které nejsou tak mokré, abychom je mohli sledovat pomocí reflexe, nebo změn intenzit odrazu světla v čase [obr. 28].

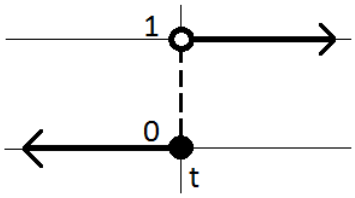
Obrázek 28: Navlhlá vozovka

Jak je možné sledovat na obrázku, je navlhlá vozovka oproti vozovce suché mnohem tmavší. Proto se nabízí jednoduchá metoda pro její zpracování a to pomocí prahování. Prahování jako takové rozděluje obraz do dvou skupin. Proto se výstup prahování často označuje jako binární obraz. Tyto dvě skupiny jsou děleny pomocí prahu t. To je hodnota, která určuje, zda bude patřit bod do jedné, nebo druhé skupiny. Je-li hodnota intenzity bodu větší nebo menší než hodnota prahu, rozhodne příslušnost k dané skupině.

Ve zpracování obrazu se většina operací navrhuje spíš pro spojité obrazy, než pro diskrétně rozdělené na jednotlivé body. A proto se také při každém návrhu musí brát ohled na to, aby se dala každá část použít vždy znovu. Většinou se s obrazem provádějí různé operace, jako jsou například derivace a jiné spojité matematické operace. Aby se však mohly provádět derivace, musí se dbát na to, aby každá funkce ve výpočtu měla v daném bodě spojitou derivaci. To ovšem není pravda pro prahování.

Obyčejné prahování $thres(x, t)$ si lze představit jako upravenou funkci signum, která pro každý vstup x snížený o t podle znaménka přiděluje hodnotu 1 nebo 0. Ovšem tato funkce není v bodě t spojitá, a nemá v ní proto derivaci (Vzorec 39, obr. 29) .

$$thres(x, t) = sgn(x - t) \tag{39}$$



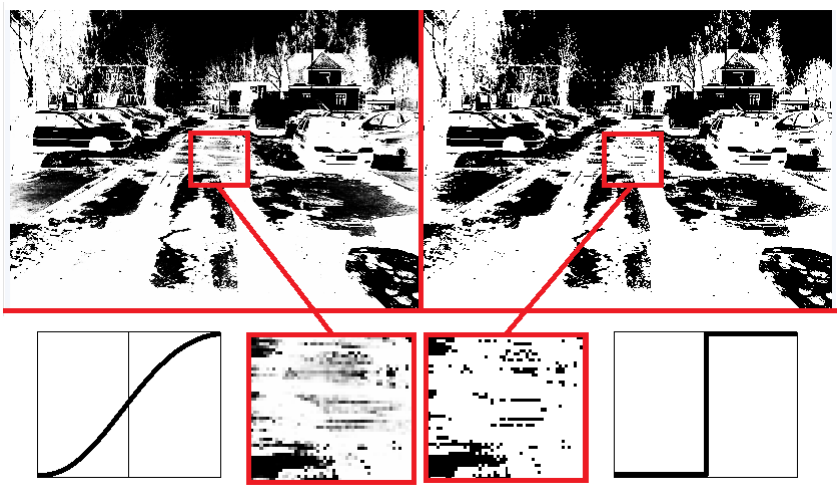
Obrázek 29: Prahování s použitím funkce signum

Při hledání odpovídající funkce, která by mohla nahradit signum, jsem narazil na sigmoidu. Sigmoida má podobný průběh jako signum, ale na rozdíl od něj je spojitá ve všech bodech, a má proto definovanou derivaci ve všech bodech. Navíc je možné si nastavit strmost λ přechodu mezi hodnotami 0 a 1. Vzorec pro prahování se sigmoidou je zde (40):

$$s(x, t, \lambda) = \frac{1}{1 + e^{-\lambda(x-t)}} \tag{40}$$

Zároveň se mi tento způsob prahování hodí pro tento modul, jelikož poskytuje mnohem hladší přechody. Je dobré si uvědomit, že integrál jakékoliv funkce reprezentující rozdělení pravděpodobnosti bude mít sigmoidní

tvar. To se dá vyložit tak, že prvky po prahování patří s určitou pravděpodobností (nebo stupněm pravdivosti) do jedné, nebo druhé skupiny. Rozdíly v prahování jsou zřejmé na následujícím obrázku [obr. 30]:



Obrázek 30: Porovnání prahování s funkcemi signum a sigmoida

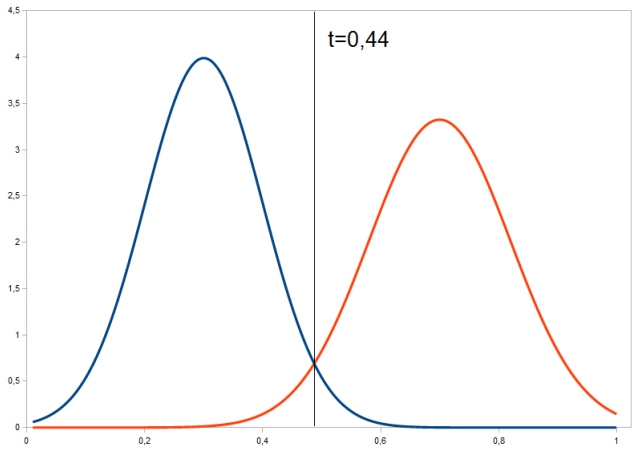
4.6.1 Pravděpodobnost a prahování

U prahování je nejdůležitější nastavení správné hodnoty prahu. Prah se nejčastěji určuje měřením. V případě výběru mokrých a suchých oblastí silnice postupujeme následovně:

1. Označíme si v obraze ručně suchou a mokrou oblast vozovky.
2. Z intenzit bodů pro mokrou a suchou vozovku vypočteme střední hodnotu, strmost a rozptyl pro normální rozdělení pravděpodobnosti. Získáme tak parametry pro dvě Gaussovy křivky.
3. Zjistíme, průsečík těchto křivek a nastavíme jej jako prah [obr. 31]. V případě, že se křivky příliš prolínají, nastavíme prah dál od té, která má menší riziko chyby. V mém případě je přijatelnější označit suchou vozovku za mokrou, protože je to bezpečnější.

4.6.2 Metoda detekce

Poté, co jsem si určil, jakým způsobem budu prahovat, je na řadě samotné nastavení prahování. Vlhká vozovka se vyznačuje několika základními rysy. Na první pohled je tmavší než vozovka suchá. Pokud se postavíme přímo proti slunci, může být odraz tak intenzivní, že bude nejsvětlejší oblastí v



Obrázek 31: Ukázka určení prahu z dvou Gaussových křivek

obraze. Tmavší než vlhká vozovka je jen velice tmavý stín. Pokud bychom měli popsat mokrou vozovku jako logický výrok, bude vypadat asi takto (41) :

$$(u \wedge \bar{v}) \vee w \tag{41}$$

u Tmavé ($x < 130$)

v Moc tmavé ($x < 30$)

w Moc světlé ($x > 250$)

Výsledkem tohoto prahování je pak opět obraz ve stupních šedi. Na následujícím obrázku jsem vedle sebe položil vstup a výstup [obr. 32].

Obrázek 32: Vstup a výstup modulu detektoru

Tmavá oblast je označená jako suchá a bílá jako mokrá. Stupeň šedi určuje stupeň pravdivosti. Hodnocení úspěšnosti detekce uvedu v kapitole měření.

4.7 Modul pro reflexi

Modul pro reflexi je určen k detekci kaluží na vozovce. Jak jsem psal již dříve, kaluž se chová na vozovce jako zrcadlo. Odráží obraz za ním položený. Abych byl schopný dokonale nalézt kaluž, musel bych znát co nejpřesnější trojrozměrnou reprezentaci sledovaného okolí. K takové rekonstrukci je třeba použít alespoň dvou kamer. Mé zadání je však omezené na jednu kameru.

Pomohl jsem si tak, že jsem trojrozměrnou reprezentaci zanedbal, protože by byl výpočet výrazně náročnější na čas a detektor by nemohl pracovat v reálném čase. Z pozorování kaluží se mi podařilo zjistit, že ve většině případů jde pouze o převrácený a mírně utlumený obraz. Ten se obrací přes horizont, který je určený y-ovou souřadnicí úběžníku. Útlum je prováděn násobením každého pixelu hodnotou menší než 1 a větší než 0. Například koeficient útlumu standardního zrcadla je kolem 0.9. Lepší zrcadla, která se vyrábějí ze stříbra, pak můžou mít tento koeficient až 0.98. Podle mého pozorování se koeficienty kaluží pohybují kolem hodnoty 0.75. Tato hodnota se hodně mění v závislosti na zakalení kaluže a aktuálního podnebí. Na následujícím obrázku ukazuji všechny kroky detekce kaluží [obr. 33].

Obrázek 33: Vstup, simulovaná reflexe a výstup modulu reflexe

Na prostředním obrázku je simulovaný odraz z kaluže. Odraz se simuluje způsobem, který jsem již naznačil. Obraz se obrátí přes horizont určený y-ovou souřadnicí úběžníku a následně se utlumí koeficientem odrazu k. Dále se pak porovnávají intenzity jednotlivých bodů obrazu a jejich rozdíl určuje chybu (vznikne obraz chyby) . Nad obrazem chyby se nakonec provádí prahování. Přesáhne-li chyba určenou hranici, není daný bod označen jako mokrá oblast vozovky.

4.8 Parametry detektoru mokré vozovky a jejich význam

Aby detektor fungoval korektně, musí se nastavit parametry, které přímo ovlivňují jeho funkci. Tyto parametry určují prahové hodnoty pro modul sledování intenzit v obraze, souřadnice úběžníku, ohniskovou vzdálenost, rozlišení zpracovávaného obrazu v modulu sledování změn intenzity odraženého světla v čase, koeficient odrazu pro modul reflexe a nakonec i logické parametry pro vyjmutí modulu ze zpracování obrazu. Jejich seznam a vysvětlení uvádím zde:

- **Vstupní obraz**
  Vstupní obraz je pořízen barevnou kamerou připevněnou v pravém horním rohu předního skla vozidla tak, aby nebránila řidiči ve výhledu. Kamera je natočena tak, aby snímala vozovku před vozidlem.

- **Šířka a výška vnitřního obrazu pro modul sledování změn intenzit odrazu v čase**
  Jelikož jsou v tomto modulu nejnáročnější operace, jako transformace obrazu na pohled na vozovku shora a zpět, detektor rohů a korespondence obrazu je vhodnější pro realtimeové zpracování pracovat s menším obrazem (100^2 nebo 200^2). Velikost tohoto obrazu výrazně ovlivňuje rychlost zpracování.

- **Souřadnice úběžníku v obraze**
  Jde o souřadnice bodu, který ukazuje, kam ubíhá silnice v obraze. Často je blízko středu obrazu.

- **Parametry Near a Far**
  Tyto parametry určují blízkou (Near) a vzdálenou (Far) hranici sledované oblasti vozovky. Hodnoty se pohybují od $0,1$ do pomyslného nekonečna (Většinou si postačím s hodnotami do $1,0$) .

- **Focus**
  Focus je parametr, jímž se nastavuje ohnisková vzdálenost pro získání pohledu na vozovku shora. Nejčastěji se hodnota ohniskové vzdálenosti pohybuje od $-0,05$ do $-0,02$.

- **ShiftX a ShiftY**
  Těmito parametry je možné posunovat pozorovanou oblast silnice. Může být užitečné, pokud se sledují jiné pruhy silnice, než ten, po kterém se vozidlo pohybuje. Většinou nechávám hodnoty nastavené na 0.

- **Parametry prahů příliš světlý, tmavý a příliš tmavý**
  Toto jsou hodnoty prahů pro modul sledování intenzity v obraze. Pro pochopení jejich významu je lepší vrátit se zpět, kde je modul popsán.

Nejčastější je nastavují na tyto hodnoty:

  - **příliš světlý** 250
  - **tmavý** 125
  - **příliš tmavý** 30

- **Koeficient odrazu**
  Koeficient odrazu určuje, jak moc se projeví odraz v kalužích na silnici. Nastavuji jej obyčejně na hodnoty od $0,65$ do $1,0$.

- **isThreshold, isReflection a isTimespace**
  Těmito parametry říkáme, zda chceme moduly sledování intenzity v obraze, reflexe a sledování změn intenzit odrazů v čase vyloučit ze zpracování, nebo je do zpracování zařadit.

  isThreshold Modul pro sledování intenzity v obraze

  isReflection Modul pro reflexi

  isTimespace Modul sledování změn intenzity odraženého světla v čase

4.9 Ukázkový zdrojový kód použití detektoru mokré vozovky

Zdrojový kód, který uvádím níže, vytváří objekt poskytovatele obrázků. Tento poskytovatel existuje ve třech verzích. Jeden načítá videosekvence, druhý snímá obraz kamerou a třetí pouze načte obrázek ze souboru. Dále se vytváří objekt detektoru mokré vozovky. Ten v cyklu získává obrázky z poskytovatele obrázků a počítá výstup. Vstup a výstup detektoru se zobrazuje ve dvou samostatných oknech.

```
1  #include "stdafx.h"
2  #include "VideoImageProvider.h"
3  #include "SimpleImageProvider.h"
4  #include "CameraImageProvider.h"
5  #include "WetRoadDetector.h"
6
7  int main(int argc, char *argv[]) {
8
9          // Připravit zdroj vstupního obrazu
10         // (Videosekvence, Obraz z kamery,
11         // Obrázek ze souboru)
12
13         // Videosekvence
14         GeneralImageProvider *imageProvider =
15                 new VideoImageProvider("cesta_new.avi");
16
17         // Kamera uEye
18         //GeneralImageProvider *imageProvider =
19         //      new CameraImageProvider(640,480);
20
21         // Obrázek ze souboru
22         //GeneralImageProvider *imageProvider =
23         //      new SimpleImageProvider("threshold.jpg");
24
25         //Kontorla - Poskytuje imageProvider obrzky?
26         if (imageProvider->image == NULL) return -1;
27         char key = 0;
28
29         // Inicializace detektoru mokré vozovky
30         WetRoadDetector *detector =
31                 new WetRoadDetector(
32                                 imageProvider,200,200,150,
33                                 150,0.3,0.1,-0.05,0,0,250,
34                                 60,30,0.9
35                         );
36
37         // vytvoření oken pro vstup a výstup
38         cvNamedWindow("video");
39         cvNamedWindow("DetectorOutput", 0);
40
41         //Pracuj, dokuď není stisknuta klávesa 'q'
42         while( key != 'q' ) {
43
44                 // Připrav další vstupní obrázek
45                 imageProvider->nextFrame();
46
47                 // Je obrázek k dispozici?
48                 if( !imageProvider->image ){
49                         break;
```

```
50                      }
51
52                      // Zpracuj aktuální snímek
53                      detector->nextFrame();
54
55                      // zobraz vstup a výstup
56                      cvShowImage("video", detector->gray);
57                      cvShowImage("DetectorOutput", detector->output);
58
59                      key = cvWaitKey(1);
60              }
61
62              // Zavři okna
63              cvDestroyWindow( "video" );
64              cvDestroyWindow( "DetectorOutput" );
65
66              // Uvolni paměť
67              delete(imageProvider);
68              delete(detector);
69
70              return 0;
71      }
```

5 Měření

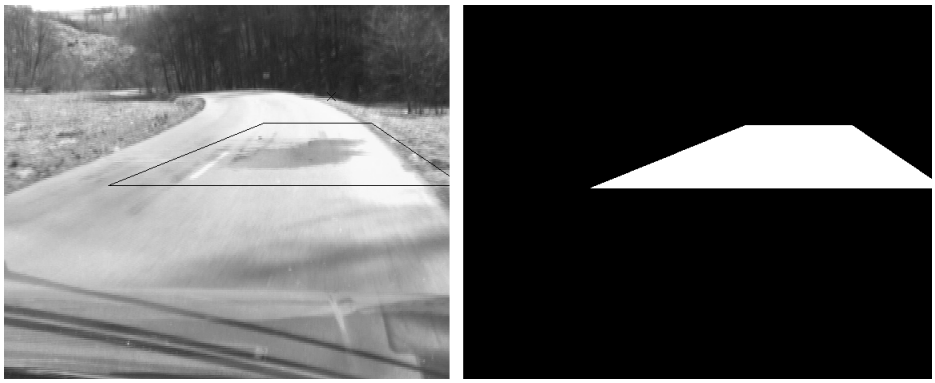
Všechna zařízení, přístroje, nebo obecně produkty se musejí řádně ověřit, než se začnou užívat v praxi. Tento detektor není vyjímkou. Měření o systému prozradí nejdůležitější informace. Z měření lze usoudit, na kolik je daný systém funkční, jak je rychlý a jestli se na něj můžeme spolehnout.

Obecně se používají k měření dva přístupy. V jednom případě se k měření používají umělé vstupy. Takové vstupy jsou však zavádějící, protože jsou přímo vytvořeny pro měřený systém. Pokud chci ověřit funkci systému lépe, musím jej testovat na reálných datech.

V mém případě je potřeba mít k dispozici videosekvence, které se získávají pomocí kamery, připevněné na předním skle automobilu. Tyto videosekvence musí být dvojího druhu. Na jedné musí být vozovka mokrá a na druhé zase suchá, aby se dala určit úspěšnost detekce v obou těchto případech.

5.1 Metoda měření a tabulka

Abych byl schopný získat dostatek údajů pro měření, musel jsem sbírat data jak v době kdy pršelo a vozovka byla mokrá, tak i za sucha. Z každého z těchto záznamů jsem vybral deset snímků, na kterých se měření provádělo (deset za sucha a deset za mokra) . Měření se provádí pouze na sledované oblasti obrazu. Pro označení aktivních bodů se používá maska [obr. 34]. Ta je stejné velikosti jako obrázek. V aktivních oblastech má hodnotu 255 a v neaktivních 0.



Obrázek 34: Maska určující aktivní body

Výstupní obraz obsahuje pixely s hodnotami od 0 do 255, které určují stupeň pravdivosti, že je vozovka mokrá. Pokud je tato hodnota rovna 0, je si detektor jistý, že je vozovka suchá. Je-li rovna 255, pak si je detektor jistý, že je vozovka mokrá.

Při testování se nejprve spočte počet aktivních bodů n_a v obraze. Při výpočtu chyby se musí myslet na to, zda je testovaná vozovka suchá nebo

mokrá. Pokud je mokrá, je procentuální chyba v daném bodě daná výrazem (42) . Protože se předpokládá u mokré vozovky, že intenzita bodu I_i výstupu detektoru má být rovna 255. Je-li intenzita menší, jde o chybu.

$$err_m(i) = \frac{255 - I_i}{255} \cdot 100[\%] \tag{42}$$

Naopak u suché vozovky se předpokládá, že bude každý bod výstupu roven 0 (není mokrá) . Výpočet procentní chyby v bode se tedy vyjádří vzorcem (43).

$$err_s(i) = \frac{I_i}{255} \cdot 100[\%] \tag{43}$$

Procentní chyba celého obrazu je dána součtem všech chyb (err_m, err_s) podělených počtem aktivních bodů n_a.

$$err_{cm} = \frac{\sum_{i=0}^{n_a} err_m(i)}{n_a} \tag{44}$$

$$err_{cs} = \frac{\sum_{i=0}^{n_a} err_s(i)}{n_a} \tag{45}$$

$$err_c = \frac{err_{cm} + err_{cs}}{2} \tag{46}$$

err_cm chyba celého obrazu mokré vozovky

err_cs chyba celého obrazu suché vozovky

err_c celková chyba

K měření jsem použil tyto přístroje:

- Notebook Acer Aspire 5920G, 2GB RAM, $2,2$GHz
- Kamera USB uEye2230-C, barevná, 1024×768

V následující tabulce [tab. 5.1] uvádím naměřenou úspěšnost detekce na jednotlivých snímcích v procentech.

Suchá vozovka	
Chyba detekce [%]	**Úspěšnost detekce [%]**
5, 85	94, 15
11, 95	88, 05
31, 86	68, 14
14, 32	85, 68
14, 61	85, 39
16, 76	83, 24
20, 88	79, 12
16, 17	83, 83
14, 61	85, 39
11, 83	88, 17
Průměrná chyba detekce [%]	**Průměrná úspěšnost detekce [%]**
15, 88	84, 12
Medián chyby detekce [%]	**Medián úspěšnosti detekce [%]**
14, 61	85, 39
Mokrá vozovka	
Chyba detekce [%]	**Úspěšnost detekce [%]**
7, 49	92, 51
70, 17	29, 83
56, 67	43, 33
65, 02	34, 98
59, 74	40, 26
28, 74	71, 26
37, 4	62, 6
18, 57	81, 43
18, 02	81, 98
5, 11	94, 89
Průměrná chyba detekce [%]	**Průměrná úspěšnost detekce [%]**
36, 69	63, 31
Medián chyby detekce [%]	**Medián úspěšnosti detekce [%]**
33, 07	66, 93
Celkové výsledky	
Průměrná chyba detekce [%]	**Průměrná úspěšnost detekce [%]**
26, 29	73, 71
Medián chyby detekce [%]	**Medián úspěšnosti detekce [%]**
17, 39	82, 61

Tabulka 1: Výsledky měření na suché a mokré vozovce

5.2 Závěry z měření

Detektor pracuje při rozlišení 640×480 v reálném čase s frekvencí kolem 10 snímků za sekundu, což znamená, že zpracování jednoho snímku mu trvá přibližně $0,1$s. Průměrná chyba měření je $26,29\%$. Úspěšnost je tedy $73,71\%$. Medián chyby je $17,39\%$ a medián úspěšnosti $82,61\%$.

Znám výsledky měření, ale je vhodné porovnat je s výsledky, kterých dosáhli lidé, pracující na podobném, nebo stejném úkolu. V publikaci [Yam05] uvádějí úspěšnost $99,4\%$ pro nalezení mokré vozovky a $75,65\%$ úspěšnosti nalezení suchých oblastí vozovky. Pokud je označena oblast obrazu jako suchá a zároveň mokrá, označují ji jako nerozhodnutelnou. Abych mohl naše výsledky porovnat, musím jejích měření převést podle následujících pravidel, aby odpovídaly mému postupu měření:

- Pokud je suchá vozovka označena jako mokrá (a naopak) , dochází v daném bodě k chybě 100%.
- Pokud je oblast označená jako nerozhodnutelná, je chyba v daném bodě 50%. To vychází z toho, že u mého výstupu je možnost získat hodnoty od 0 (není mokrá) do 255 (je mokrá) . V případě, že je hodnota rovna 127 patří stejnou měrou do množiny suché i mokré vozovky. To odpovídá zmíněné chybě 50%.

Výpočet probíhá následovně:

Nejprve vypočtu chybu pro mokrou vozovku. Jelikož udávají pouze úspěšnost, musím ji odečíst od 100 a dostanu chybu pro mokrou vozovku.

$$100 - 99,4 = 0,6[\%] \tag{47}$$

Chyba pro suchou vozovku se počítá obdobně.

$$100 - 75,65 = 24,35[\%] \tag{48}$$

Dále je třeba zjistit, kolik procent oblastí je detekovaných jako nerozhodnutelných. Když znám chyby detekce suché a mokré vozovky, tak mohu usoudit, že jejich rozdíl je nerozhodnutelná oblast.

$$24,35 - 0,6 = 23,75[\%] \tag{49}$$

Celkovou průměrnou chybu pak mohu vypočíst váženým průměrem podle určených pravidel.

$$err = \frac{(1 \cdot 0,6 + 1 \cdot 24,35 + 0.5 \cdot 23,75)}{3} = 12,275[\%] \tag{50}$$

Po převodu vychází jejich úspěšnost na $87,75\%$ což je o $14,015\%$ lepší než u mého detektoru. Přesto toto porovnání není příliš dobrým ukazatelem. Abych mohl porovnat oba detektory přesněji, musel bych je oba zkoušet na stejných testovacích datech. A navíc bohužel nevím nic o vzhledu a povrchové úpravě silnic v Japonsku.

V mém detektoru nejčastěji nastaly chyby tam, kde se objevil stín a modul pro sledování intenzit v obraze jej detekoval jako mokrou vozovku. Podobnost těchto oblastí je velká a jejich rozlišení je tedy velice náročné. Přiklonil jsem se k variantě detekce stínu jako mokré vozovky, protože je to lepší z hlediska bezpečnosti. Raději budu detekovat stín jako mokrou vozovku, než mokrou vozovku jako suchou a zastíněnou.

Dále docházelo k falešným detekcím ze strany modulu pro reflexi, který našel odrazy tam, kde nebyly. Objevovaly se stejné intenzity bodů ve spodní i horní části obrazu. Docházelo k tomu hlavně v oblastí rohů a hran, kde se mění pomalu intenzita bodů.

Modul pro sledování změn intenzit odrazu zase někdy špatně odhadl posun obrazu (rychlost) , a vytvořil tak falešnou změnu intenzity odrazu, která vedla k falešné detekci. To nastávalo hlavně v místech, kde je v obrazech velká změna intenzity odstínu šedé (derivace).

Převážně dochází k chybám vlivem rohů a hran v obraze. Napadlo mě, že bych oblasti kolem hran a rohů utlumoval, ale při detekci by došlo k častějším záměnám mokrých oblastí za suché. To je nepřijatelné, a proto jsem výstup ponechal i s těmito chybami.

Pokud by byl výstupní obraz složen z binárních hodnot, mělo by smysl navíc filtrovat oblasti s menším počtem pixelů, protože jde s největší pravděpodobností pouze o nějakou nečistotu na vozovce, nebo šum.

Obrázek 35: Vstup a výstup detektoru

6 Závěr

Úkolem mé práce bylo navrhnout a implementovat systém pro detekci mokré vozovky z jedoucího vozidla na základě analýzy obrazu z jediné kamery. Je to jedna z významných úloh asistenčních systémů v dopravě. Systém je možno použít ve prospěch řidiče, nebo případně do informačního systému, ze kterého mohou i ostatní řidiči čerpat užitečné informace.

K detekci se užívá kamera uEye2230-C připevněná v pravém horním rohu předního skla řidiče, kde nebrání ve výhledu a snímá vozovku před vozidlem. Tato kamera je připojena k počítači, na kterém je implementovaný detektor. Implementaci detektoru jsem provedl v jazyce C++ a s výhodou využil knihovnu OpenCV, určenou ke zpracování obrazu.

Z dlouhodobého pozorování vozovky za sucha i za mokra jsem identifikoval čtyři typy vozovky: suchá, navlhlá, mokrá a kaluž. Suchá vozovka se vyznačuje tím, že při jejím pozorování z různých úhlů pohledu se intenzita odraženého světla nemění a je světle šedá. Navlhlá vozovka má tmavší barvu, avšak intenzita odrazu se při pozorování z různých úhlů pohledu také nemění. Mokrá vozovka je tmavší, a však při změně úhlu pohledu již lze pozorovat změny intenzity odraženého světla. Kaluž se chová jako zrcadlo položené na vozovku a můžeme v něm pozorovat obrácený a mírně utlumený obraz prostředí za ní.

Na základě těchto pozorování jsem navrhl systém, který využívá tři moduly: Modul sledování změn intenzity odraženého světla v čase, Modul pro sledování intenzit v obraze a Modul pro reflexi (Odraz) . Modul sledování změn intenzity odraženého světla v čase je určen pro detekci mokré vozovky. Pomocí homogenní transformace získá obraz jejího povrchu (Pohled na vozovku shora) a s využitím korespondence obrazu sleduje, v za sebou jdoucích snímcích, změny intenzity odraženého světla. Modul pro sledování intenzit v obraze je určen k sledování vlhké vozovky a pracuje na základě prahování, které jsem upravil pro tento účel z binárních hodnot na Fuzzy logické hodnoty, umožňující plynulé přechody. Modul pro reflexi pracuje na základě simulace zrcadlení vozovky. Obrátí se obraz přes horizont a porovnává se se sledovanou oblastí vozovky. Pokud je podobnost vysoká, označí se oblast jako kaluž.

Po implementaci jsem systém řádně experimentálně testoval na reálných datech, která jsem získal snímáním vozovky za sucha i za mokra. Z těchto snímků jsem některé náhodně vybral pro testování. Výsledkem je naměřená průměrná úspěšnost $73,71\%$. To je o $14,015\%$ méně, než výsledek po přepočtu, který se uvádí v textu [Yam05]. Detekce navíc probíhá v reálném čase s frekvencí kolem deseti snímků za sekundu, při rozlišení vstupního obrazu 640×480.

Práce by se měla dále ubírat směrem k filtraci nežádoucích jevů, jako jsou například: kapky na skle před kamerou, vliv přímého slunečního záření do kamery a překrytí výhledu stěračem. Dále je pak možné využít obrazu

povrchu vozovky a sledovat na něm orientační prvky, ty rozpoznávat a informovat o nich řidiče.

Při řešení této úlohy jsem se seznámil s mnoha odbornými články, a rozšířil jsem si tak znalosti získané v předmětech Analýza a zpracování obrazu a nabyl některé nové. Za hlavní výhodu považuji to, že jsem si veškeré nabyté znalosti hned prakticky vyzkoušel a naučil se je používat v praxi.

Literatura

[Soj03] Eduard Sojka *Počítačová grafika II: Metody a nástroje zobrazování 3D scén*,
Ostrava: VŠB TUO, RCCV. 2003. ISBN 80-248-0293-7

[Soj00] Eduard Sojka *Digitální zpracování a analýza obrazu*,
VŠB-TU Ostrava, 2000

[She06] Mohamed Shehata, Muzamil Pervez, Jun Cai, Wael Badawy, Ahmad Radmanesh: *Real Time Static Glare Identification in ITS*, 2.5.2006
Department of Electrical and Computer Engineering, University of Calgary, 2500 University Drive, N.W., Calgary, Alberta, Canada T2N 1N4
Shoubra Faculty of Engineering, Zagazig University, 108 Shoubra Street, Shoubra, Cairo, Egypt
The City of Calgary, Transportation Dept./Signals Division, P.O. BOX 2100, Stn. M, #4009, Calgary, Alberta, Canada T2P 2M5

[Tes08] Tomoaki Teshima, Yuko Uematsu, Hideo Saito, Masayoshi Shimizuz: *Wet/Dry classification method with reflect and incident light observed from single camera.* 26.10.2008
School of Science and Technology, Keio University 3-14-1 Hiyoshi Yokohama Kanagawa 223-8522, Japan
Fujitsu Laboratories Ltd 4-1-1 Kamikodanaka Kawasaki Kanagawa 211-8588, Japan

[Tes07] Tomoaki TESHIMA, Yuko UEMATSU, and Hideo SAITO *Detection of The Wet Area of The Road Surface Based on a Saturated Reflection.* 22.6.2007
Graduate School of Science and Technology, Keio University 3-14-1 Hiyoshi, Kohoku-ku 223-8522, Japan

[Ran09] A. L. Rankin, L. H. Matthies *Daytime Mud Detection for Unmanned Ground Vehicle Aautonomous Navigation.* 7.7.2009
Jet Propulsion Laboratory, California Institute of Technology 4800 Oak Grove Drive, Pasadena, CA, USA 91109

[Yam05] Muneo Yamada, Koji Ueda, Isao Horiba, Shin Yamamoto, and Sadayuki Tsugawa *Detection of Wet-Road Conditions from Images Captured by a Vehicle-Mounted Camera.* 20.4.2005
Journal of Robotics and Mechatronics Vol.17 No.3, 2005

[Jok07] Maria Jokela, Matti Kutila, Jukka Laitinen, Florian Ahlers, Nicolas Hautičre, and Tobias Schendzielorz *Optical Road Monitoring of the*

Future Smart Roads – Preliminary Results. 24.7.2007
World Academy of Science, Engineering and Technology 34 2007

[Tes09] Tomoaki Teshima, Hideo Saito, Masayoshi Shimizu, Akinori Taguchi *Classification of Wet/Dry Area Based on the Mahalanobis Distance of Feature from Time Space Image Analysis*. 20.5.2009
MVA2009 IAPR Conference on Machine Vision Applications, May 20-22, 2009, Yokohama, JAPAN

[Har88] Chris Harris, Mike Stephens *A Combined Corner and Edge Detector*,
Plessey Research Roke Manor, United Kingdom © The Plessey Company pic. 1988

[Mor80] Moravec H., *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*,
Tech Report CMU-RI-TR-3, Carnegie-Mellon University, Robotics Institute, September 1980

[Wel06] Greg Welch,Gary Bishop *An Introduction to the Kalman Filter*,
Department of Computer Science University of North Carolina at Chapel Hill Chapel Hill, NC 27599-3175 Monday, July 24, 2006